* Cookie的弊端：cookie虽然在持久保存客户端数据提供了方便，分担了服务器存储的负担，但还是有很多局限性的。
  + 每个特定的域名下最多生成20个cookie
  + IE和Opera 会清理近期最少使用的cookie，Firefox会随机清理cookie。
  + cookie的最大大约为4096字节，为了兼容性，一般不能超过4095字节。
* 优点：极高的扩展性和可用性
  + 通过良好的编程，控制保存在cookie中的session对象的大小。
  + 通过加密和安全传输技术（SSL），减少cookie被破解的可能性。
  + 只在cookie中存放不敏感数据，即使被盗也不会有重大损失。
  + 控制cookie的生命期，使之不会永远有效。偷盗者很可能拿到一个过期的cookie。
* 缺点：
  + `Cookie`数量和长度的限制。每个domain最多只能有20条cookie，每个cookie长度不能超过4KB，否则会被截掉。
  + 安全性问题。如果cookie被人拦截了，那人就可以取得所有的session信息。即使加密也与事无补，因为拦截者并不需要知道cookie的意义，他只要原样转发cookie就可以达到目的了。
  + 有些状态不可能保存在客户端。例如，为了防止重复提交表单，我们需要在服务器端保存一个计数器。如果我们把这个计数器保存在客户端，那么它起不到任何作用。
* 浏览器本地存储：
  + 在较高版本的浏览器中，js提供了sessionStorage和globalStorage。在HTML5中提供了localStorage来取代globalStorage。
  + html5中的Web Storage包括了两种存储方式：sessionStorage和localStorage。
  + sessionStorage用于本地存储一个会话（session）中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储。
  + 而localStorage用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。
* web storage和cookie的区别：
  + Web Storage的概念和cookie相似，区别是它是为了更大容量存储设计的。Cookie的大小是受限的，并且每次你请求一个新的页面的时候Cookie都会被发送过去，这样无形中浪费了带宽，另外cookie还需要指定作用域，不可以跨域调用。
  + 除此之外，Web Storage拥有setItem,getItem,removeItem,clear等方法，不像cookie需要前端开发者自己封装setCookie，getCookie。
  + 但是Cookie也是不可以或缺的：Cookie的作用是与服务器进行交互，作为HTTP规范的一部分而存在 ，而Web Storage仅仅是为了在本地“存储”数据而生
  + 浏览器的支持除了IE７及以下不支持外，其他标准浏览器都完全支持(ie及FF需在web服务器里运行)，值得一提的是IE总是办好事，例如IE7、IE6中的UserData其实就是javascript本地存储的解决方案。通过简单的代码封装可以统一到所有的浏览器都支持web storage。
  + localStorage和sessionStorage都具有相同的操作方法，例如setItem、getItem和removeItem等。